

TURING 图灵程序设计丛书

Apress®

Exploring C++
The Programmer's Introduction to C++

C++探秘

68讲贯通C++

[美] Ray Lischner 著
刘晓娜 林健 译
石小兵 李杰

- 步步为营攻破C++堡垒
- 编程专家为你指点迷津
- C++初学者进阶必备

 人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

Exploring C++
The Programmer's Introduction to C++

C++探秘

68讲贯通C++

人民邮电出版社
北京

图书在版编目 (C I P) 数据

C++探秘 : 68讲贯通C++ / (美) 里斯纳
(Lischner, R.) 著 ; 刘晓娜等译. — 北京 : 人民邮电
出版社, 2011.1

(图灵程序设计丛书)

书名原文: Exploring C++: The Programmer's
Introduction to C++
ISBN 978-7-115-24227-3

I. ①C… II. ①里… ②刘… III. ①C语言—程序设
计 IV. ①TP312

中国版本图书馆CIP数据核字(2010)第239061号

内 容 提 要

本书采用引导探索式的教学方法, 将庞大的 C++ 知识体系划分成四个大部分 68 讲, 每讲都包含一个互动练习, 帮助读者循序渐进地学习 C++。你可以通过这种互动快速掌握表达式、声明、标准库、自定义函数、类和模板等等 C++ 的各方面特性, 并最终掌握如何把这些特性组合起来编写复杂的 C++ 程序。

本书适合有少量其他语言编程经验的 C++ 初学者。

图灵程序设计丛书

C++探秘：68讲贯通C++

- ◆ 著 [美] Ray Lischner
译 刘晓娜 林 健 石小兵 李 杰
责任编辑 明永玲
执行编辑 丁晓昀
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
中国铁道出版社印刷厂印刷
 - ◆ 开本: 800×1000 1/16
印张: 36.5
字数: 862千字 2011年1月第1版
印数: 1-3 000册 2011年1月北京第1次印刷
- 著作权合同登记号 图字: 01-2007-5618号
ISBN 978-7-115-24227-3
-

定价: 89.00元

读者服务热线: (010)51095186 印装质量热线: (010)67129223

反盗版热线: (010)67171154

版 权 声 明

Original English language edition, entitled *Exploring C++: The Programmer's Introduction to C++* by Ray Lischner, published by Apress, 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705 USA.

Copyright © 2009 by Ray Lischner.

Simplified Chinese language edition copyright © 2011 by Posts & Telecom Press. All rights reserved.

本书中文简体字版由Apress L.P.授权人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

前 言

大家好，首先谢谢你们阅读我的书。我叫 Ray，是本书的作者。从现在开始我们将要一起相处很长一段时间，因此先拉把椅子，舒服地坐下来吧。我的任务就是帮你学习 C++，因此我编写了一系列的课程，每节课（即每讲）都包含一个互动练习，帮助你循序渐进地学习 C++。你的任务是完成每讲的阅读和练习，通过这种方法来学习 C++。

你肯定已经大致翻阅过本书。如果没有，现在就浏览一下。请注意，本书和大部分其他书籍都不同。大部分编程书籍只不过是些书面讲稿，作者告诉你知识点，并期望你去阅读、学习和理解这些知识。

本书则截然不同。因为我觉得照本宣科没什么意义，而且照本宣科并不是最好的教学方式。本书中，你将通过阅读、修改和编写程序来学习编程。为此，我组织书的方式就是让你花尽量多的时间阅读、修改和编写程序。

如何使用本书

本书的每一讲都既有文字又有交互式练习。这些练习和你在其他书上看到的不一样。我的作业中没有多选、填空或者简答题，都是对 C++ 关键特性的交互式探索。在本书的前一部分，我会给出一个完整的程序。随着学习的深入，你将可以修改和扩展该程序。很快，你就能自己编写整个程序了。

“交互”的意思是，我提出问题，你回答问题，我尽量在行文中点评你的回答。听起来很疯狂吧？但是通过回答问题，你能更好地学习 C++。为此，我在书中留出填写答案的空白，你可以把答案写在书上（除非你使用的是从朋友或者从图书馆借来的书）。实际上，我鼓励你把所有的答案都写在书上。只有通过回答问题，你才能真正地学好这门课。

有的问题没有唯一正确的答案。我提出问题是让你思考，让你从一个新的角度去看待一个问题。大部分问题都有清晰正确的答案，而我会在今后的叙述中给出答案，因此不要向后跳读。你要先写下答案，然后再去验证。有些问题可能很棘手或者需要我还没有讲到的知识，这种情况下，我估计你的答案可能会是错的，但是错了也没有关系。不要担心，我不会给你打分。（如果本书被用作正式的教材，老师应该只依据是否完成这些练习来打分，而不用管答案是否正确。老师应该另有其他的练习、测验来评估学生们学习这门课程的效果。）不要偷看后面的内容，

去抄“正确”的答案，这样做你将学不到任何东西。

准备好了吗？开始热身吧。

阅读本书时，你最重要的任务是什么？

这个问题没有唯一正确的答案，但确实有很多明显错误的答案。希望你的答案类似于“完成每个练习”或者“理解所有的知识点”。“享受其中的乐趣”也是个好答案。

本书的结构

C++是一门复杂的语言，即便编写最简单的程序，也要理解该语言的很多不同方面。这门语言本身难以划分为几个一般性的主题，比如函数、类、语句或者表达式。因此本书没有尝试这样去组织结构，而是让你一点点增量式地学习 C++：学一点这个，学一点那个，再多学一点这个，很快你就能积累足够的知识，可以开始编写复杂的程序了。

粗略地讲，本书以基本表达式、声明和语句开始，学会这些就足以编写简单的程序。在本书的前几讲中，你会学到如何使用标准库。然后再学习如何编写自己的函数、类和模板，再然后编写复杂一些的程序。

但是，阅读完本书后，你也不会立即成为专家。你还需要更多的练习，从广度和深度上理解这门语言和库，再不断实践。你需要不断地实践，明白了吧？

本书的读者对象

如果你有兴趣学习 C++且至少了解一门其他的编程语言，则可以阅读本书。具体地说，你不需要了解 C，也不需要了解面向对象编程的知识。

C 程序设计语言影响了许多其他语言的设计，比如 PHP、Perl、AWK、C#当然还有 C++。因此，尽管许多程序员不了解 C 和 C++，但他们仍然会发现很多语言结构似曾相识，有些人甚至觉得可以跳过本书讲解背景的几讲。但是我不建议这样做！因为我会先讲解一些 C++独有的特性。在少数情况下，我会告诉你什么时候可以放心地跳过某节，而且仅仅只是跳过该节。即使你很熟悉某个语言特性，但该特性在 C++中也可能会有不一样的地方。

对于 C 程序员，固有思维模式是很危险的，虽然表面上 C++和 C 极为相似，但是实际上 C 程序员需要克服的问题最多。从设计上说，很多 C 程序也是有效的 C++程序，这就导致粗心的 C 程序员误以为“好的 C 程序也是好的 C++程序”。实际上，C 和 C++是不同的语言，各自有不同的习语和特质。要成为一名高效的 C++程序员，你必须学会 C++的编程方式。C 程序员需要突破已经形成的习惯，学会弃用某些 C 特性（比如数组），转而使用更好的 C++习语。本书的结构能帮助你以 C++（而非 C）的方式思考问题。

项目

本书还包括 4 个项目，可以利用这些项目巩固所学的知识。每个项目都有特定的价值，可以充分运用引入该项目之前所学的 C++ 知识。我鼓励你尝试每个项目，用你最拿手的软件设计技术完成这些项目。记住，除了编写源码外还要编写测试用例，要尽量让代码在保证正确的基础上清晰易读。完成你的方案之后，从本书网站 (<http://cpphelp.com/exploring/>) 下载相应文件，并把你的方案和我的进行对比。

合作

你可以独自一人使用本书自学 C++，或由老师指导，用本书作为正式的授课教材。你还可以和同伴一起学习，这样会更有乐趣，而且通过合作，你也能学得更多更快。此书需人手一册，阅读完每一课的内容并独立完成任务。如果碰到问题，可以和你的同伴讨论，但是要独立完成练习，然后将你的答案和同伴的作比较。如果你们的答案不同，可开展讨论，陈述各自的理由，看能否达成一致的见解。

请和同伴合作完成书中的项目。你可以将工作分成两个以上的模块，或者一个人负责编码，另一个人负责检查，你们也可以实践一下结对编程。总之，怎么适合就怎么来，但是要确保你理解项目中的每一行代码。如果你们扮演不同的角色，请在不同的项目中互换角色，让每个人都有机会做多方面的事。

参考资料

本书不能教你关于 C++ 的所有知识，实际上没有一本书可以做到这一点。阅读完本书后，还应该继续阅读和编写大量 C++ 程序，并寻找其他的信息来源。本书有一个专门的指导网站 <http://cpphelp.com/exploring/>。该网站上有其他书、其他网站、邮件列表、讨论组、常见问题、编译器和其他工具等的链接。你也可以从网站上下载本书所有的源代码，以节省打字时间。

为何叫 C++ 探秘

如果你觉得本书有点儿怪异，那敬请放心，“这些虽是疯话，却有深意在內”。^①

这层深意，就是把我俄勒冈州立大学教计算机时所积累的东西诉诸文字，奉献给广大读者。为了提高教学质量，我特意研究了知识获取的过程，尤其是科学知识及计算机编程。

通过这几十年的研究，我总结出这样一点：每个人都在意识中构建了客观世界的一个模型，而获取新知识的过程就是向这个模型中不断地注入新的信息，且新的信息必须与这种模型相容。但有时候，新信息与原有模型冲突，则必须要调整我们的模型以容纳新的信息。我们的大脑，就是在不断地工作，不断地得到新的信息，而不断地调整意识中的模型。

^① 原文为 “though this be madness, yet there is method in’t.”，出自《哈姆雷特》。——译者注

这种研究的结果是，课堂的重心由教师转移到了学生身上。在过去，教师把学生当作是一个个空的容器，等着教师用知识与智慧来填充，学生都是被动地接受信息。但是现在我们明白了，学生不是被动的接受者，他们是活跃的，尽管有时他们显得木讷而被动，不过他们的大脑每时每刻都在运转，都在接受新的信息并把它们纳入到自己的思维模型中。而教师则不再是自诩的智慧之源，而是要作为这种思维模型的引导员。并且教师不能直接地去改造他们的模型，而仅是创造一种课堂环境，让学生有机会去调整他们自己的模型。

尽管这个研究主要针对教师，但对于作者，又何尝不是如此呢？

因此，我不会教你 C++，而是在创造一种探索的氛围，让读者在其中去学习 C++。尽管这种探索模式并不是研究学习和写作的唯一方式，但它确实是通过我多年来的教学总结及完善得来，并成功运用于实践当中的，原因有如下几点。

- 它让你不得不主动地融入到学习过程中。被动地读书当然轻松，而不断提出问题则能促使你去寻找解决方案，从而让它们成为你的思维模型中的新部分。但如果你跳过这些问题，那就同时跳过了完善思维模型的机会。
- 探索过程比较短，这使你的模型能够循序渐进地不断完善。假如每次都要填塞大量的新信息，则很可能导致你的模型中夹杂错误的信息。一旦这种错误信息根深蒂固，则更加难以改正。因此我需要保证你的思维模式随时都保持尽可能精确。
- 这些探索构建在所学内容的基础上。我不会抛出一堆新概念，并指望你能一下子掌握它们，而是尽力把这些新概念联系到某些已学过的概念，保证这些概念可以牢牢地印在你的思维模型中。
- 它帮助你从实践找到真知。每一讲的核心并不在于去咀嚼别人的解决方案，而是让你尽量把时间用在自主解决问题的“刀刃”上：修改原有程序，编写新程序。

C++是一门复杂的语言，而学习 C++也绝非易事。在任一组 C++程序员中，即便一个简单的问题都会引出不同的答案。对于大部分 C++程序员而言，他们的语言思维模型不仅不完整，而且有缺陷，甚至还有根本性的问题。我希望能够提供给大家一个坚实的 C++基础，从而让你能写出正确而有趣的程序，更重要的是，让你可以在未来的学习中能继续享受 C++的美妙和乐趣。

C++标准

本书涵盖了当前的标准，即 ISO/IEC 14882:2003(E)，*Programming languages—C++*。2003 版的标准是一个修订版，包括对 1998 版的修订和澄清。大部分的现代编译器遵循 2003 版的标准。

标准化委员会还发行了一份该标准的附录，添加了正则表达式、数学函数等许多内容。这份附录是标准库的可选扩展项，称为 Technical Report 1 (技术报告 1)，或者 TR1。因为它是可选的，所以不要求厂商必须实现它。大部分厂商提供了该库的部分实现。一些厂商实现了完整的 TR1。使用本书并不需要 TR1 的支持，但是我在书中指出了一些情况，在这些情况下 TR1 能使得编码更容易。

通过发行 TR1 以及让成千上万的 C++开发人员使用它，标准化委员会获得了宝贵的实践经验，并将其反馈到 C++标准的下一次主要的修订中。在我编写本书的时候，下一次修订也正在进行。你阅读本书时，修订工作可能已经完成，甚至可能有了符合新标准的编译器和库。新标准可能会被标记为 ISO/IEC 14882:2010(E)。

即使有了全新的编译器，本书仍然有价值。许多新特性是高级特性，因此不会影响本书。其他一些计划中的特性恐怕就会影响所有的 C++程序员。在本书中，我指出了有可能发生变化之处，但同时本书主要关注当前可用的和广泛使用的工具。

致 谢

本书比以前我编写的所有书籍都难写，因而更耗时。我对帮助我完成这项工程的 Apress 公司及其员工的谢意难以言表，尤其是 Matthew Moodie 和 Richard Dal Porto，他们在整个过程中表现出极大的耐心。

我尤其要感谢我的技术审稿人 Francis Glassborow，他对 C++ 的深刻理解帮助我避免了很多大大小小的错误。如果仍然存在技术错误，肯定是在编辑流程后期由我引入的，才会逃过了 Francis 的注意。

我最想要感谢的人是我的妻子 Cheryl。当我后劲不足难以为继时，是她的支持和鼓励让我坚持下去。我也要感谢我的几个孩子，在我编写此书期间无法陪伴他们度过许多日夜。我爱你们。

最后，我要感谢在治疗风湿性关节炎方面创造奇迹的科学家和医生们，是你们让我能够继续工作、写作和娱乐。

目 录

第一部分 C++基础

第 1 讲 打磨工具	2	7.1.1 初始化	37
1.1 作者推荐	2	7.1.2 条件	38
1.1.1 Windows 平台	2	7.1.3 后循环	39
1.1.2 Macintosh OS 9 以及更早版本	3	7.1.4 for 循环的工作原理	39
1.1.3 其他平台	3	7.2 由你来做	39
1.2 阅读文档	3	第 8 讲 格式化输出	41
1.3 第一个程序	4	8.1 问题	41
第 2 讲 阅读 C++代码	10	8.2 字段宽度	42
2.1 注释	11	8.3 填充	43
2.2 头文件	11	8.4 std 前缀	44
2.3 主程序	13	8.5 对齐	44
2.4 变量定义	13	8.6 探索格式化	44
2.5 语句	14	8.7 替代语法	46
2.6 输出	15	8.8 由你完成	46
第 3 讲 整数表达式	17	第 9 讲 数组和向量	49
第 4 讲 字符串	23	9.1 用向量代替数组	49
第 5 讲 简单的输入	28	9.2 向量	50
第 6 讲 错误消息	33	9.3 迭代器	51
6.1 拼写错误	34	9.4 算法	53
6.2 错误字符	34	9.5 成员类型	54
6.3 未知操作符	35	9.6 使用迭代器和算法	55
6.4 未知名字	35	第 10 讲 自增和自减	58
6.5 符号错误	36	10.1 自增	58
6.6 从错误中获得乐趣	36	10.2 自减	59
第 7 讲 For 循环	37	第 11 讲 条件和逻辑	64
7.1 有界循环	37	11.1 I/O 和 bool	64
		11.2 布尔类型	65
		11.3 逻辑操作符	67
		11.4 旧式语法	68

11.5 比较操作符	68	20.3 常量引用	119
第 12 讲 复合语句	71	20.4 const_iterator	120
12.1 语句	71	20.5 输出参数	121
12.2 局部定义和范围	74	第 21 讲 使用算法	122
12.3 for 循环头中的定义	76	21.1 传递数据	122
第 13 讲 文件 I/O 简介	79	21.2 谓词	126
13.1 读文件	79	21.3 其他算法	128
13.2 写文件	80	第 22 讲 重载函数名	131
第 14 讲 数据结构映射	83	第 23 讲 大数和小数	136
14.1 使用映射	83	23.1 长整型和短整型	136
14.2 迭代器	84	23.1.1 长整数	137
14.3 搜索映射	86	23.1.2 短整数	137
第 15 讲 类型同义词	88	23.2 整数字面量	138
15.1 typedef 声明	88	23.3 字节长度的整数	139
15.2 常见的类型定义	89	23.4 类型转换	140
第 16 讲 字符	91	23.5 整数算术	141
16.1 字符类型	91	23.6 重载解析	142
16.2 字符 I/O	93	第 24 讲 极大数和极小数	145
16.3 换行和移植性	94	24.1 浮点数	145
16.4 转义字符	94	24.2 浮点字面量	146
第 17 讲 字符分类	96	24.3 浮点特征	147
17.1 字符集	96	24.4 浮点 I/O	148
17.2 字符分类	98	第 25 讲 文档	151
17.3 区域设置	99	25.1 Doxygen	151
第 18 讲 大小写转换	103	25.2 结构化注释	151
18.1 简单的大小写	103	25.3 文档标签	152
18.2 复杂的大小写	104	25.4 使用 Doxygen	156
第 19 讲 编写函数	107	第 26 讲 项目 1: 身体质量指数	157
19.1 函数	107	第二部分 自定义类型	
19.2 函数调用	109	第 27 讲 自定义类型	160
19.3 声明和定义	109	27.1 定义新类型	160
19.4 再谈单词计数	111	27.2 成员函数	161
19.5 函数 main()	113	27.3 构造函数	164
第 20 讲 函数实参	115	27.4 重载构造函数	166
20.1 实参传递	115	第 28 讲 重载操作符	167
20.2 按引用传递	117	28.1 比较有理数	167

28.2 算术操作符	171	36.2 虚函数	229
28.3 数学函数	173	36.3 引用与切除	230
第 29 讲 自定义 I/O 操作符	175	36.4 纯虚函数	231
29.1 输入操作符	175	36.5 虚析构函数	232
29.2 输出操作符	176	第 37 讲 类与类型	233
29.3 错误状态	177	37.1 类与 typedef	233
第 30 讲 赋值与初始化	179	37.2 值类型	236
30.1 赋值操作符	179	37.2.1 复制	236
30.2 构造函数	180	37.2.2 赋值	236
30.3 合并	181	37.2.3 比较	236
第 31 讲 编写类	186	37.3 资源获取即为初始化	239
31.1 类的结构	186	第 38 讲 声明与定义	241
31.2 成员函数	187	38.1 声明与定义	241
31.3 构造函数	189	38.2 内联函数	243
第 32 讲 深入探索成员函数	193	38.3 变量声明与定义	244
32.1 调用默认构造函数	193	38.4 静态变量	246
32.2 重温 Project 1	196	38.5 静态数据成员	248
32.3 const 成员函数	199	38.6 声明符	250
第 33 讲 访问级别	203	第 39 讲 使用多个源文件	251
33.1 公有与私有	203	39.1 多个源文件	251
33.2 class 与 struct	206	39.2 声明与定义	252
33.3 简单的旧式数据	206	39.3 #include 文件	254
33.4 公有还是私有	207	39.3.1 引号与括号	256
第 34 讲 面向对象编程介绍	212	39.3.2 嵌套#include 指令	256
34.1 书籍与杂志	212	39.3.3 包含监护	257
34.2 分类	213	39.3.4 文档	258
34.3 继承	215	39.4 外部变量	261
34.4 Liskov 置换原则	216	39.5 内联函数	261
34.5 类型多态	216	39.6 “一份定义”规则	262
第 35 讲 继承	218	第 40 讲 函数对象	264
35.1 派生类	218	40.1 函数调用操作符	264
35.2 析构函数	221	40.2 函数对象	265
35.3 访问级别	224	40.3 复用成员函数	269
35.4 编程风格	225	40.4 生成器函子	270
第 36 讲 虚函数	226	第 41 讲 有用的算法	272
36.1 类型多态	226	41.1 搜索	272
		41.1.1 线性搜索算法	272
		41.1.2 二分搜索算法	276
		41.2 比较	279

41.3 重组数据	281	46.6 声明与定义	331
41.4 复制数据	282	46.7 成员函数模板	331
41.5 删除元素	283		
41.6 迭代器	284	第 47 讲 类模板	333
第 42 讲 迭代器	285	47.1 参数化类型	333
42.1 迭代器的种类	285	47.2 参数化 rational 类	334
42.1.1 输入迭代器	286	47.3 使用类模板	336
42.1.2 输出迭代器	286	47.4 重载的操作符函数	338
42.1.3 前向迭代器	286	47.5 混合类型	340
42.1.4 双向迭代器	287		
42.1.5 随机访问迭代器	287	第 48 讲 模板特化	342
42.2 使用迭代器工作	288	48.1 实例化与特化	342
42.3 const_iterator 与 const iterator	290	48.2 自定义比较函数	345
42.4 错误消息	292	48.3 特化函数模板	346
42.5 专用迭代器	292	48.4 特征	347
42.6 迭代器要点	294		
第 43 讲 异常	296	第 49 讲 部分特化	349
43.1 异常介绍	296	49.1 退化的 pair	349
43.2 捕获异常	297	49.2 部分特化	350
43.3 抛出异常	299	49.3 部分特化函数模板	351
43.4 程序栈	300	49.4 值模板形参	351
43.5 标准异常	304		
43.6 I/O 异常	304	第 50 讲 名字与名字空间	353
43.7 自定义异常	306	50.1 名字空间	353
43.8 对异常的建议	307	50.2 嵌套名字空间	355
第 44 讲 更多操作符	309	50.3 全局名字空间	358
44.1 条件操作符	309	50.4 名字空间 std	358
44.2 短路操作符	311	50.5 使用名字空间	359
44.3 逗号操作符	311	50.5.1 using 指令	359
44.4 算术赋值操作符	313	50.5.2 using 声明	361
44.5 自增与自减	315	50.5.3 类中的 using 声明	363
		50.6 无名名字空间	364
		50.7 名字查找	365
第 45 讲 项目 2: 定点数	318		
		第 51 讲 容器	370
		51.1 容器的性质	370
		51.2 技术报告 1	371
		51.3 成员类型	372
		51.4 容器里能放什么	373
		51.5 插入与清除	374
		51.5.1 顺序容器的插入操作	374
		51.5.2 顺序容器的清除操作	375
		51.5.3 关联容器的插入操作	375
		51.5.4 关联容器的清除操作	376
		51.5.5 异常	377
第三部分 泛型编程			
第 46 讲 函数模板	324		
46.1 泛型函数	324		
46.2 使用函数模板	325		
46.3 编写函数模板	326		
46.4 模板形参	328		
46.5 模板实参	329		

51.6	迭代器与引用	377	57.4	实现标准容器	440
51.7	顺序容器	380	57.5	增加变量	441
51.7.1	类模板 array	381	57.6	特殊成员函数	448
51.7.2	类模板 deque	382			
51.7.3	类模板 list	383	第 58 讲 异常-安全		452
51.7.4	类模板 vector	384	58.1	内存泄漏	452
51.8	关联容器	385	58.2	异常与动态内存	454
			58.3	自动删除指针	456
第 52 讲 国际字符		389	58.4	auto_ptr 不能做的事	458
52.1	为何要“宽”	389	58.5	异常与构造函数	458
52.2	使用宽字符	389			
52.3	宽字符串	390	第 59 讲 旧式数组		462
52.4	宽字符的 I/O 操作	392	59.1	C 风格的数组	462
52.5	多字节字符集	393	59.2	数组的限制	463
52.6	Unicode	394	59.3	动态分配数组	464
52.7	通用字符名字	396	59.4	多维数组	465
			59.5	C 风格的字符串	466
第 53 讲 区域设置与分面		397	59.6	命令行参数	466
53.1	问题	397	59.7	指针运算	468
53.2	拯救者“区域设置”	398			
53.3	区域设置与 I/O	399	第 60 讲 智能指针		470
53.4	分面	399	60.1	重新审视 auto_ptr	470
53.5	字符类别	402	60.2	可复制智能指针	472
53.6	排序规则	406	60.3	智能数组	474
			60.4	Pimpl	474
第 54 讲 文本 I/O		410	60.5	迭代器	482
54.1	文件模式	410			
54.2	字符串流	411	第 61 讲 位操作		483
54.3	文本转换	417	61.1	将整数作为位的集合	483
54.4	Boost 词法转换	420	61.2	位掩码	485
			61.3	移位	486
第 55 讲 项目 3: 货币类型		422	61.4	使用无符号类型安全移位	487
			61.4.1	有符号与无符号类型	488
			61.4.2	无符号字面量	488
			61.4.3	类型转换	489
			61.5	溢出	493
			61.6	位域简介	493
			61.7	可移植性	494
			61.8	bitset 类模板	495
			第 62 讲 枚举		498
			62.1	理想的枚举	498
			62.2	作为位掩码的枚举	499

第四部分 实时编程

第 56 讲 指针		424
56.1	问题	424
56.2	解决方案	432
56.3	地址与指针	433
56.4	依赖图	434
第 57 讲 动态内存		437
57.1	分配内存	437
57.2	释放内存	438
57.3	指向为空的指针	438

62.3 模拟枚举.....	500	64.4 基于策略的编程.....	530
62.3.1 枚举计算机语言.....	500	第 65 讲 名字与模板	538
62.3.2 对语言进行比较.....	501	65.1 限定名的问题.....	538
62.3.3 赋值.....	502	65.2 非限定名的问题.....	540
62.3.4 字符串和语言.....	503	第 66 讲 重载函数	546
62.3.5 初始化.....	508	66.1 重载函数回顾.....	546
62.3.6 读写语言.....	508	66.2 重载解析.....	549
62.3.7 使用模拟的枚举.....	509	66.2.1 候选函数.....	549
62.4 重新审视项目.....	510	66.2.2 可行函数.....	551
第 63 讲 多重继承	512	66.2.3 最佳可行函数.....	551
63.1 多重基类.....	512	66.3 默认实参.....	555
63.2 虚基类.....	514	第 67 讲 元编程	557
63.3 类 Java 接口.....	516	67.1 编译时编程.....	557
63.4 接口与模板.....	518	67.2 模板特化.....	557
63.5 Mix-in.....	519	67.3 部分特化.....	559
63.6 友元来帮忙.....	521	第 68 讲 项目 4: 计算器	566
第 64 讲 特征萃取与策略	524		
64.1 案例研究: 迭代器.....	524		
64.2 迭代器特征萃取.....	528		
64.3 案例研究: char_traits.....	529		



在探索 C++ 之前，首先要准备一些基本的工具：文本编辑器、C++ 编译器、链接器和调试器。你可以从不同的地方分别获取这些工具，也可以获取一个包含所有这些工具的软件包，例如一个 IDE（Integrated Development Environment，集成开发环境）。不管你使用什么平台、操作系统以及有多少预算，你都有很多种选择。

如果你是在课堂上学习 C++，老师会提供这些工具或者指定使用哪些工具。如果你在一个使用 C++ 的组织中工作，可能需要使用他们的工具，以便熟悉并恰当地使用这些工具。如果你要获取自己的工具，可以从本书网站 (<http://cpphelp.com/exploring/>) 下载。工具的版本和特性变化太快，本书可能并没有对这些工具提供详细说明，但是可以在网站上找到最新信息。接下来的一讲将给出一些通用建议。

1.1 作者推荐

C++ 是世界上应用最为广泛的编程语言之一，仅次于 C（这取决于如何定义“广泛应用”）。因此，很多硬件和软件环境中都有对应的 C++ 工具，但它们的价格高低不一。

你可以选择命令行工具，命令行工具在 UNIX 和类 UNIX 环境中尤其流行；或者选择一个把所有工具都集成在图形用户界面（GUI）下的 IDE。只要你觉得方便，什么样的工具其实都无所谓，程序不会关心你用什么工具去编辑、编译和链接它们。

1.1.1 Windows 平台

如果你在使用微软的 Windows 平台，我推荐微软的 Visual Studio（我指的是当前最新版本）。特别要注意的是，容易受攻击的 Visual C++ 6.0 已经过时了。在我写作本书时，最新版本是 Visual Studio 2008。如果你不想花钱，可以从微软网站下载 Visual C++ Express（可以在 <http://cpphelp.com/> 上面找到当前版本的链接）。如果想使用开源软件，可以下载 MinGW，它是将流行的 GNU 编译器移植到 Windows 平台的一个版本。

注意 C++/CLI 与 C++ 不是一回事，前者是微软发明的一种新语言，目的是将 C++ 集成到 .NET 环境中。别看它的名字包含 C++，正如 C++ 这个名字源自于 C 但却不是 C 一样，C++/CLI 也不是 C++。本书的内容只包含标准 C++，不会涉及其他内容。如果你决定使用 Visual Studio，注意